[2]:

```
def dfs(graph, node, visited=None, result=None):
    if visited is None:
        visited = set()
    if result is None:
        result = []

    if node not in visited:
        visited.add(node)
        result.append(node)  # Store output in a list

        for neighbor in graph[node]:
            dfs(graph, neighbor, visited, result)

    return result  # Return the list of visited nodes

# Example graph represented as an adjacency list
graph = {
    'A': ['B', 'C'],
    'B': ['D', 'E'],
    'C': ['F'],
    'D': [],
    'E': ['F'],
    'F': []
}
```

jupyterlite depth first search Last Checkpoint: 5 minutes ago

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python (Pyodide)

```
graph = {
    'A': ['B', 'C'],
    'B': ['D', 'E'],
    'C': ['F'],
    'D': [],
    'E': ['F'],
    'F': []
}

dfs_result = dfs(graph, 'A')  # Start DFS from node 'A'
print("DFS Traversal as List:", dfs_result)
```

```
DFS Traversal as List: ['A', 'B', 'D', 'E', 'F', 'C']
```